REMARKS

This is response to the Notice of Non-Compliant Amendment mailed on January 24, 2008 and further in response to the Office Action mailed on July 13, 2007. Applicant is hereby re-submitting the same amendment sent in previously on October 12, 2007 with the proper identifiers and the text of the canceled claim 2 not appearing in the amendments to claims section. No other revisions have been made and no new matter is presented. Therefore, Applicant believes everything outlined in the Notice of Non-Compliance has been remedied and consideration of the remarks to the Office Action are respectfully requested.

In this Office Action, claims 16-25 were rejected under 35 USC §101 and claims 1-25 were rejected under 35 U.S.C. §102. All three independent claims, 1, 16, 25, were therefore rejected in this Office Action. In this Response, claims 1, 14 and 16 have been amended and claim 2 has been cancelled.

REJECTIONS UNDER 35 U.S.C. §101

In the Office Action, claims 16-25 were rejected under 35 U.S.C. §101 because the claims were said to be directed to non-statutory subject matter. In particular, claims 16-20 were said to define a non-statutory subject matter because they were asserted to be software per se. Claims 21-25 were rejected as being directed to a computer program as well.

1. THE LAW OF PATENTABLE SUBJECT MATTER

§101 extends the offer of patent protection to "any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof". As Congress commented in passing the statute, it was intended to cover "anything under the sun that was made by man", and the Supreme Court and the Court of Appeals for the Federal Circuit have both reiterated that observation, along with noting that the repeated usage of the word "any" applied to expansive descriptions of subject matter, were intended to emphasize that no restrictions were to be placed on patentable

subject matter other than those specifically recited in §101. (S. Rep. No. 1979, 82d Congress, 2d Sess., 5 (1952); *Diamond v. Chakrabarty*, 447 U.S. 303, 206 USPQ 193 (1980); *State Street Bank & Trust v. Signature Financial Group*, 47 USPQ2d 1596, 1600 (Fed. Cir. 1998) (Rich, J.).)

Claims directed to methods have been found to be within the "process" category of §101. *AT&T Corp. v. Excel Communications, Inc.*, 50 USPQ2d 1447, 1450 (Fed. Cir. 1999) and MPEP §2106 IV A. Claims with functional material recorded on a computer-readable medium have been found to be within the "machine" category of § 101 and are considered statutory. *In re Lowry*, 32 USPQ2d 1031, 1035 (Fed. Cir. 1994) and MPEP §2106.01.

Despite the seemingly limitless expanse to patentable subject matter, the Supreme Court has identified three categories of unpatentable subject matter: "laws of nature, natural phenomena, and abstract ideas." *Diamond v. Diehr*, 450 U.S. 175, 185, 209 USPQ 1 (1981) However, determining what is an "abstract idea" has been difficult for the courts. As noted by the U.S. Court of Appeals for the Federal Circuit "this court (and its predecessor) has struggled to make our understanding of the scope of §101 responsive to the needs of the modern world." *AT&T Corp. v. Excel Communications, Inc.*, 50 USPQ2d 1447, 1452 (Fed. Cir. 1999)

In *AT&T*, the Federal Circuit gave some guidance by stating that "the mere fact that a claimed invention involves inputting numbers, calculating numbers, outputting numbers, and storing numbers, in and of itself, would not render it nonstatutory subject matter, unless, of course, its operation does not produce a 'useful, concrete and tangible result.'" *AT&T* at 1453. The formation of a 'useful, concrete and tangible result' in a claim constitutes a practical application of a mathematical algorithm, formula, or calculation and is therefore patentable subject matter. *State Street* at 1601.

Similarly, the Interim Guidelines for Examination of Patent Applications for Patent Subject Matter Eligibility as found in MPEP 2106 (IV)(C) indicates that abstract ideas are not patentable subject matter under 35 U.S.C. §101 but that "practical applications" of abstract ideas are patentable subject matter. A claimed invention is a practical application of an abstract idea if either:

"The claimed invention 'transforms' an article or physical object to a different state or thing." or

"The claimed invention otherwise produces a useful, concrete and tangible result, based on the factors discussed below." (MPEP 2106(IV)(C)(2)

Note that the case law and the Interim Guidelines do not require a claimed use for the "useful, concrete and tangible result". They only require the production of a "useful, concrete and tangible result."

The question then becomes: What constitutes a "useful, concrete and tangible result?"

The term "useful" appears in §101 and requires nothing more than a specific, substantial and credible utility. (MPEP 2107.01)

The terms "concrete" and "tangible" have not been directly defined. However, a review of recent cases that have attempted to use this standard is instructive.

The phrase "useful, concrete, and tangible result" first appears in *In re Alappat*, 31 USPQ2d 1545 (Fed. Cir. 1994). The claims at issue in *Alappat* were directed to a rasterizer that included as a last limitation "means for outputting illumination intensity data as a predetermined function of the normalized vertical distance." Thus, the "result" in *Alappat* is "illumination intensity data", which was considered concrete and tangible. This data is nothing more than numbers that represent a specific intensity level for light that may appear on a display at some point in the future. Nonetheless, the data were considered patentable subject matter.

In *State Street Bank & Trust v. Signature Financial Group*, 47 USPQ2d 1596, 1600 (Fed. Cir. 1998) , the Federal Circuit built on the "useful, concrete and tangible result" test by finding that a data processing system that produces "price, profit, percentage, cost, or loss" provides a useful, concrete and tangible result even though these values are expressed as numbers. *State Street Bank* at 1602. The numbers identified as a useful, concrete and tangible result merely represent the state of an accounting system for a mutual fund. In addition, the claims in *State Street Bank* do not recite a use for these values. Thus, the values produced were considered "useful, concrete and tangible" even without a claimed use for those values.

Lastly, in *AT&T Corp. v. Excel Communications, Inc.*, 50 USPQ2d 1447, (Fed. Cir. 1999), the Federal Circuit applied the "useful, concrete and tangible result" test to a method claim. In *AT&T,* a method is provided for generating a message record that includes a PIC indicator. This message record represents a call made on a telephone system. The message record was considered a "useful, concrete and tangible result" even though it is only a collection of data representing a telephone call. In addition, as in *State Street Bank*, the claims in *AT&T* did not include a use for the result of the method. In particular, the claim in *AT&T* simply claims producing a message record not a use for that message record. Thus, a method claim does not need to include a use for a result of the method but instead simply has to generate a "useful, concrete and tangible result" in order to be considered a practical application of the method.

2. APPLICATION OF THE LAW TO THE CLAIMS

CLAIMS 16-20

Independent claim 16 has been amended to include "a *computer implemented* system for developing software applications" (emphasis added). This claim provides a computer-implemented system for developing software applications in a managed code execution environment using a string resource tool. The string resource tool supplements the design program and enables a developer to verify that a resource identifier is correctly addressed to a string and key name so as to correspond to a managed code resource that is supported by the managed code execution environment.

The computer system of claim 16 provides a practical application because it produces a "useful, concrete and tangible result." In particular, the system is implemented on at least one computer to develop software applications, and therefore must produce a "useful, concrete and tangible result."

As indicated on page 15, lines 11-25, a computer system having such a string resource tool enables a programmer to quickly have convenient design-time accessibility to resource address/index information and also optionally to the values that correspond to catalogued resources. This can be displayed in a pop-up window and/or a drop-down list within the design program interface to assist the developer in statement and function completion. This reduces the amount of coding errors prior to runtime (pg. 3, lines 20-

23). As such, the string resource tool has a specific and credible utility in programming. Therefore, the result of claim 16 is a useful result.

A computer system having a string resource tool also has a concrete and tangible result. A computer system having such a string resource tool is at least as concrete and tangible as profit and loss data as found in *State Street Bank*, a message record as found in *AT&T*, and intensity data as found in *Alappat*. Such a string resource tool does not exist as an abstraction but instead is stored on a computer system from which it can be used in programming. In addition, the string resource tool is a real-world tool that must be created in order for certain coding errors to be detected before runtime. The fact that the string resource tool is needed for certain error detection prior to runtime indicates that it is not a mere abstraction but a concrete and tangible thing. Since the string resource tool provided by claim 16 is a useful, concrete and tangible result, claim 16 provides a practical application of its method. As such, claim 16 represents statutory subject matter under the current case law and the MPEP.

CLAIMS 21-25

Independent claim 21 provides a string resource tool for reducing coding errors prior to runtime. The string resource tool is used in the context of a managed code execution environment by providing a tool component that provides string information. The string information enables a developer to select, from a closed set of alternatives, a particular identifier that represents a string.

A string resource tool that reduces coding errors prior to runtime is a useful, concrete and tangible result. As such claim 21 provides a practical application of its method.

As indicated above, a string resource tool is useful in programming because it can be used to reduce the number of errors at runtime that must be debugged.

A string resource tool in a managed code execution environment is also a concrete and tangible result because it is at least as concrete and tangible as the profit and loss data of *State Street Bank*, the message record of *AT&T*, and the intensity data of *Alappat*. Since the string resource tool provided by claim 21 is a useful, concrete and tangible result, claim 21 provides a practical application of its method. As such, claim 21 represents statutory subject matter under the current case law and the MPEP.

REJECTIONS UNDER 35 USC 102

The Office Action rejected claims 1-25 under 102(b) as being anticipated by Craig Utley, "A Programmer's Introduction to Visual Basic .NET", SAMS Publishing, 2001 (hereinafter: Utley). As discusssed below, it is respectfully submitted that claims 1-25 are now patentably distinguishable from Utley.

CLAIMS 1-15

Applicant's claim 1 has been amended to include the limitations from claim 2, namely to include: "verifying that a resource identifier input by the developer corresponds to one of the plurality of managed code resources by: providing the developer with a collection of resource identifiers; and receiving said resource identifier input from the developer in the form of a selection from the collection of resource identifiers." The verification occurs when the developer inputs a resource identifier during code development. When the developer does begin to input a resource identifier, a collection of already existing resource identifiers are given to the developer. The developer can then verify the resource exists before inputting it into their code (page 18, line 25 -- page 19, line 24).

The Office Action indicates on pg. 4 first paragraph, that Utley on p. 142 teaches section validation controls for verification. The verification taught in Utley occurs on the client side (pg. 142-143). This is distinguishable from the current application where verification occurs when the developer, not the client, inputs a resource identifier.

The Office Action also asserts that figure 6.13, pg. 116 and figure 6.14 pg. 119, figure 6.15, providing binding data, string collections also teaches the above claim. Figure 6.13 shows a string collection editor open with the developer inputting names into the editor (pg. 116, para. 2). Figure 6.14 shows the developer testing the project after the strings have been inputted. There is a drop down box and the strings the developer has put in are available if the project is successful(pg. 119, para. 3). Figure 6.15 shows a form with a searchable index. Fig. 6.15 shows that the index comprises visual basic documentation, such as functions (else, else if, end if, etc. are listed in index).

Respectfully these references do not teach the applicant's claim 1. In the above reference, nowhere does it teach or recite a method of verifying a resource

identifier's existence, based upon the developer's input during programming. The developer generates the strings of figures 6.13 and 6.14. These strings are not selected by the developer from a selection of resource identifiers to verify the programmer's input exists. The index of fig. 6.15 does provide the programmer with the capability to look up functions, but does not check a programmer's input against resources to verify their existence.

CLAIMS 16-20

Claim 16 is amended in this response to include "a string resource tool that supplements the design program and enables a developer to verify that a resource identifier is correctly addressed *to a string and key name* so as to correspond to a managed code resource that is supported by the managed code execution environment" (emphasis added).

Claim 16 has a string resource tool that verifies that a resource identifier is correctly addressed to a sting and key name so as to correspond to a managed code resource. The access to resources can be displayed in pop-up windows and/or drop-down lists (e.g., directly within the design program interface) to assist the developer in statement and function completion. For example, identifier information (i.e., key name, string and/or value information) that corresponds to available resources is displayed to the developer for selection and statement completion (Pg. 15, para. 2).

The Office Action indicated claim 16 is anticipated by the frameworks shown in pages 26, 32, 35 etc. and Figure 6.13 pg. 116 and figure 6.14, pg. 119, and figure 6.15 having a text box that is accessible to code resources. The applicant respectfully traverses this argument in light of the following. In none of the above citations to the reference does it teach to verify that a resource identifier is correctly addressed to a sting and key name so as to correspond to a managed code resource supported by the execution environment. The framework of page 26 shows a solution explorer window, but does not mention verification of a key word and string. Similarly, the framework on page 32 illustrates a code window but does not show verification of a key word and string. On page 35 a framework for debugging is shown, but does not illustrate a specific string resource tool

that supplements a design program and enables a developer to verify that a resource identifier is correctly addressed to a string and key name. Instead, it shows a debugging window with debugging statements. The developer generates the strings of figures 6.13 and 6.14. These strings are not selected by the developer from a selection of resource identifiers to verify their existence. The index of fig. 6.15 does provide the programmer with the capability to look up functions, but does not check a programmer's input against resources and does not mention or display a key word and string verification. It is therefore respectfully submitted that cited references do not teach or suggest claim 16.

CLAIMS 21-25

Independent claim 21 was also rejected as being anticipated by Utley in the frameworks shown in pages 26, 32, 35 etc. and Figure 6.13 pg. 116 and figure 6.14, pg. 119, and figure 6.15 having a text box that is accessible to code resources. The applicant respectfully traverses this argument in light of the following.

Claim 21 is directed to a resource tool that provides string information through a design program interface, wherein the string information enables a developer to select from a closed set of alternative a particular identifier that represents a particular string.

In none of the above citations to the reference does it teach a resource tool that provides string information through a design program interface. The above citations further do not teach that the string information enables a developer to select from a closed set of alternatives, a particular identifier that represents a string. The framework of page 26 shows a solution explorer window, but does not mention providing string information or selecting an identifier from a closed set of alternatives that represents a string. Similarly, the framework on page 32 illustrates a code window but does not mention providing string information or selecting an identifier from a closed set of alternatives that represents a string. On page 35 a framework for debugging is shown, but does not mention providing string information or selecting an identifier from a closed set of alternatives that represents a string. Instead, it shows a debugging window with debugging statements. The developer generates the strings of figures 6.13 and 6.14. These strings are not selected by the developer from a selection of resource identifiers to verify their existence during programming time. The index of fig. 6.15 does provide the programmer with the

capability to look up functions, but does not discuss or illustrate using string information to enable a developer to select from a closed set of alternatives a particular identifier that represents a string. It is therefore respectfully submitted that the cited references do not teach or suggest claim independent claim 21.

CONCLUSION

It is respectfully submitted claims 16-25 are directed to statutory subject matter. It is also respectfully submitted that claims 1-25 are novel and non-obvious relative to the cited prior art. It is therefore respectfully that submitted claims 1-25 are in form for allowance. Reconsideration and allowance of the claims are respectfully requested.

The Director is authorized to charge any fee deficiency required by this paper or credit any overpayment to Deposit Account No. 23-1123.

Respectfully submitted,

WESTMAN, CHAMPLIN & KELLY, P.A.

By: _____

Christopher L. Holt, Reg. No. 45,844
900 Second Avenue South, Suite 1400
Minneapolis, Minnesota 55402-3319
Phone: (612) 334-3222 Fax: (612) 334-3312

CLH:CRC:NB:rkp